Integrating Rotations using Non-Unit Quaternions

Caleb Rucker *Member, IEEE*

***Abstract*—In this paper, we propose and demonstrate a method for integration of rotations using non-unit quaternions. Unit quaternions are commonly used to represent rotation, in which case the rotation operation involves the conjugate of the unit quaternion. However, a redundant mapping can be defined from all nonzero quaternions to the set of rotation matrices, $\mathrm{SO}(3)$, based on the more general rotation operation involving the quaternion inverse. From this we show that the well-known formula which maps angular velocity to the derivative of a unit quaternion actually represents the minimum-norm solution within a set of solutions for the derivative of a non-unit quaternion. This fact enables efficient, singularity-free, numerical integration of rotations over long intervals. The approach inherently preserves the structure of $\mathrm{SO}(3)$ during integration with any standard routine or ODE solver package without employing specially designed geometric integration schemes, exponential updates, or the many quaternion length enforcement techniques found in the literature. We demonstrate the accuracy of this approach compared to other common methods applied to integrate a known angular velocity function and a classic Lagrange top.**

***Index Terms*—Simulation and Animation, Kinematics, Dynamics**

I. Introduction: Representing Rotation

In rigid-body dynamics, robotics, and other related fields it is widely recognized that there are fundamental trade-offs in the choice of representations for rotations [1]. The special orthogonal group $\mathrm{SO}(3) = \{\mathbf{R} \in \mathbb{R}^{3\times 3} \mid \mathbf{R}^T\mathbf{R} = \mathbf{I} \text{ and } |\mathbf{R}| = 1\}$ (the set of rotation matrices) is often viewed as the most convenient representation in robotics because the rotation operation is simply matrix-vector multiplication, and the matrix columns form an orthonormal basis which can be viewed as the axes of a Cartesian reference frame attached to a moving rigid body. Further, given an angular velocity vector expressed in either the body frame or spatial frame, the derivative of a rotation matrix $\mathbf{R} \in \mathrm{SO}(3)$ is linear in its own elements [2]:

$$\dot{\mathbf{R}} = \mathbf{R}[\boldsymbol{\omega}_b]_\times \qquad \dot{\mathbf{R}} = [\boldsymbol{\omega}_s]_\times \mathbf{R} \tag{1}$$

where $\boldsymbol{\omega}_b \in \mathbb{R}^3$ and $\boldsymbol{\omega}_s \in \mathbb{R}^3$ are the angular velocity vectors expressed in body-frame and spatial coordinates respectively, and $[\,]_\times$ denotes the standard mapping from $\mathbb{R}^3$ to $\mathfrak{so}(3)$ (the 3×3 skew-symmetric matrices) as

$$[\boldsymbol{\omega}]_\times = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \tag{2}$$

Unfortunately, direct numerical integration of the above differential equations with standard vector-space integration routines such as explicit Runge-Kutta methods does not preserve the structure of $\mathrm{SO}(3)$, even for constant angular velocities. The columns of $\mathbf{R}$ become non orthogonal and even non-unit length due to truncation and roundoff error inherent to the numerical integration routine at every step. These errors can be acceptably small if a high-order routine is used with a small enough step size over a short integration length, but the structural error continually compounds such that integration over long intervals becomes problematic.

The problem of maintaining the structure of $\mathrm{SO}(3)$ is an example of a larger class of problems addressed by the field of geometric numerical integration, in which numerical methods are designed to preserve geometric properties of the flow of a differential equation, such as invariance of the Hamiltonian in conservative systems [3]. For the rotation problem, these methods are either implicit, requiring approximate solution of a nonlinear system at each integration step, or they are formulated by making updates via a mapping that preserves $\mathrm{SO}(3)$, such as the exponential map (e.g. Crouch-Grossman methods) [4]–[7]. The simplest exponential update is to hold the angular velocity constant over a time step and apply the matrix exponential solution to the linearized differential equation. Exponential updates require a substitution at the identity to avoid dividing by zero, and higher-order exponential methods are more difficult to derive and implement.

Another common approach to the problem is to recognize that a rotation fundamentally has only three degrees of freedom and employ a three-parameter representation of rotation which maps $\mathbb{R}^3$ to $\mathrm{SO}(3)$, such as any of the Euler angle or Tait-Bryan angle sets. Computationally, this guarantees $\mathbf{R}$ is orthonormal, but all such three-parameter representations contain singularities, i.e. there are some orientations for which the mapping from angular velocity to Euler angle derivatives is undefined due to a division by zero. For Euler angles this is the well-known phenomenon of gimbal lock. Even for rotations close to these singularities, the required Euler angle derivatives may be so large as to degrade the accuracy of numerical integration. Less well-known three-parameter representations include Rodrigues parameters, Wiener–Milenkovic parameters, and exponential coordinates via the Euler–Rodrigues formula [1] (which are used to develop Munthe-Kaas geometric integration methods [4], [6], [8]).

Since all three-parameter representations suffer from singu-

Manuscript received: February, 23, 2018; Revised May, 10, 2018; Accepted June, 8, 2018.

This paper was recommended for publication by Editor Nancy Amato upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by the National Science Foundation under CAREER Award IIS-1652588 and under CMMI-1427122 as part of the NSF/NASA/NIH/USDA/DOD National Robotics Initiative. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

Caleb Rucker is with the University of Tennessee, Knoxville, TN 37996, USA caleb.rucker@utk.edu

Digital Object Identifier (DOI): see top of this page.

larities, it is natural to consider four-parameter representations with a constraint to eliminate the redundancy. Unit quaternions are a four parameter representation with no representational singularities. They are favored in computer graphics and aerospace applications for their computational efficiency (no trigonometric functions are required, and the differential equations are linear) and their natural adaptation to interpolation of discrete rotations. Orientation estimation filters based on unit quaternions are an active area of research in industrial manipulation [9], human motion tracking [10]–[12], and unmanned aerial vehicles [13], [14].

The conventional wisdom in fields which use quaternions to represent rotation is that one must take efforts to ensure that the quaternion is always unit length so that the redundancy of representing a 3-dimensional rotation with a set of four parameters is resolved (up to the sign of the quaternion since unit quaternions provide a double cover of $\mathrm{SO}(3)$). Thus the unit length is viewed as a constraint that must be enforced alongside the system of ODEs that governs the quaternion evolution. Hence, a differential-algebraic-equations approach is taken in [15], a predictor-corrector scheme in [16], [17], integration based on exponential updates in [7], [9], [12]–[14], a Munthe-Kaas-like method on the quaternion's exponential generator in [18], and step-wise normalization in [7], [10], [11] where the quaternion state variables are overwritten with their normalized values after each integration step.

A. Contributions and Related Work

The contribution of this paper is to derive and demonstrate a method for integrating rotations using non-unit quaternions, thus avoiding the necessity of enforcing unit quaternion length as an algebraic constraint, which is often recognized as the main trade-off of unit quaternions [4]. We briefly discussed this method in our recent work on real-time elastic rod dynamics [19]. Recent work by Boyle [8] has also recognized this possibility and applied it to an astrophysical problem. In [20], Tunay used non-unit quaternion length to parameterize strain energy associated with inflation in elastic rods.

Based on the general formula for rotation using non-unit quaternions, we construct a straightforward mapping from the nonzero quaternions to rotation matrices. We then derive the set of solutions to the under-determined system for the derivatives of the non-unit quaternion elements. The conventional evolution equation for unit quaternions belongs to this set and turns out to be the minimum-norm solution for the derivative of a non-unit quaternion. We then discuss how this fact enables the use of any high-order integration scheme or general-purpose ODE solver to efficiently integrate rotations over long time periods while eliminating singularities and maintaining the structure of $\mathrm{SO}(3)$. We explore the nuanced relationship between the non-unit approach and step-wise normalization, and we compare various integration routines on non-unit quaternions with two different exponential update methods. Finally, we compare various rotation parameterizations applied to the integration of a Lagrange top. Results demonstrate that the non-unit method performs comparably to step-wise renormalization, and slight accuracy gains may be seen in some problems.

II. Rotations using Non-Unit Quaternions

A full exposition of quaternions can be found in several places, e.g. [21]. Quaternions were discovered by William Rowan Hamilton as an extension of the complex numbers to form a non-commutative algebra in four dimensions. A quaternion

$$\mathbf{q} = q_0 + q_1 i + q_2 j + q_3 k \tag{3}$$

consists of a scalar part q_0 and a vector part $q_1 i + q_2 j + q_3 k$ where $i,\ j,\ k$ are the so-called quaternionic units. Multiplication of two quaternions can be succinctly defined by the law

$$i^2 = j^2 = k^2 = ijk = -1 \tag{4}$$

which Hamilton is famously said to have carved into the Brougham bridge. For the purposes of this paper, we will also treat quaternions as vectors in $\mathbb{R}^4$

$$\mathbf{q} = [q_0\ q_1\ q_2\ q_3]^T \tag{5}$$

in which case the quaternion product $\mathbf{qp}$ of two quaternions $\mathbf{q}$ and $\mathbf{p}$ is equivalent to a matrix-vector product

$$\mathbf{qp} = \mathbf{Q}(\mathbf{q})\mathbf{p} \tag{6}$$

where

$$\mathbf{Q}(\mathbf{q}) = \begin{bmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & q_3 & -q_2 \\ q_2 & -q_3 & q_0 & q_1 \\ q_3 & q_2 & -q_1 & q_0 \end{bmatrix} \tag{7}$$

The conjugate of a quaternion $\mathbf{q}$ is obtained by negating the vector part

$$\mathbf{q}^* = [q_0\ -q_1\ -q_2\ -q_3]^T. \tag{8}$$

The identity quaternion is $[1\ 0\ 0\ 0]^T$, and the inverse of a nonzero quaternion is given by the conjugate divided by the norm squared

$$\mathbf{q}^{-1} = \frac{\mathbf{q}^*}{\|\mathbf{q}\|^2}, \qquad \|\mathbf{q}\|^2 = q_0^2 + q_1^2 + q_2^2 + q_3^2 = \mathbf{q}^T\mathbf{q} \tag{9}$$

The set of nonzero quaternions forms a group under quaternion multiplication with these identity and inverse definitions. A vector $\mathbf{v} \in \mathbb{R}^3$ can be considered a quaternion with zero scalar part (a so-called pure quaternion). If $||\mathbf{q}|| \neq 0$, the triple quaternion product

$$\mathbf{v}' = \mathbf{q}\mathbf{v}\mathbf{q}^{-1} \tag{10}$$

then produces another pure quaternion $\mathbf{v}'$ which is a rotational transformation of $\mathbf{v}$. The quaternion can be written in terms of the angle of rotation θ, the unit vector of the axis of rotation $\hat{\mathbf{n}}$, and an arbitrary nonzero scalar factor α as

$$\mathbf{q} = \alpha \begin{bmatrix} \cos\frac{\theta}{2} \\ \hat{\mathbf{n}} \sin\frac{\theta}{2} \end{bmatrix}. \tag{11}$$

Due to the inverse in the rotation formula, the factor α does not influence the result of the rotation operation, and thus all nonzero scalar multiples of a quaternion correspond to the same rotation. The rotation defined by the triple product above is equivalent to the matrix-vector product

$$\mathbf{v}' = \mathbf{R}\mathbf{v} \tag{12}$$

where

$$\mathbf{R} = \frac{1}{\|\mathbf{q}\|^2}\begin{bmatrix} q_0^2+q_1^2-q_2^2-q_3^2 & 2(q_1q_2-q_0q_3) & 2(q_1q_3+q_0q_2) \\ 2(q_1q_2+q_0q_3) & q_0^2+q_2^2-q_1^2-q_3^2 & 2(q_2q_3-q_0q_1) \\ 2(q_1q_3-q_0q_2) & 2(q_2q_3+q_0q_1) & q_0^2+q_3^2-q_1^2-q_2^2 \end{bmatrix} \quad (13)$$

This equation maps any nonzero quaternion to a rotation matrix $\mathbf{R} \in \mathrm{SO}(3)$.

Clearly, there is a simplification that could be made by only considering quaternions with $||\mathbf{q}|| = 1$ (unit quaternions). Denoting $\hat{\mathbf{q}}$ as a unit quaternion, the inverse and rotation operations become

$$\hat{\mathbf{q}}^{-1} = \hat{\mathbf{q}}^* \qquad \mathbf{v}' = \hat{\mathbf{q}}\mathbf{v}\hat{\mathbf{q}}^* \quad (14)$$

and the rotation matrix formulation is the same as (13) without the factor $\frac{1}{\|\mathbf{q}\|^2}$ in front.

$$\mathbf{R} = \begin{bmatrix} q_0^2+q_1^2-q_2^2-q_3^2 & 2(q_1q_2-q_0q_3) & 2(q_1q_3+q_0q_2) \\ 2(q_1q_2+q_0q_3) & q_0^2+q_2^2-q_1^2-q_3^2 & 2(q_2q_3-q_0q_1) \\ 2(q_1q_3-q_0q_2) & 2(q_2q_3+q_0q_1) & q_0^2+q_3^2-q_1^2-q_2^2 \end{bmatrix} \quad (15)$$

The set of unit quaternions forms a Lie group (a group which is also a smooth manifold where the multiplication and inverse operations are smooth) that double covers the rotation group $\mathrm{SO}(3)$ since $-\mathbf{q}$ corresponds to the same rotation as $\mathbf{q}$, thus eliminating most of the redundancy of representing 3D rotation with four parameters. For these reasons, the convention is that only *unit* quaternions are used to represent rotation. However, as reviewed in the introduction, this comes at the price of having to treat the unit length as a constraint and formulate a strategy for reducing or eliminating constraint drift over time.

We suggest that a better strategy is to use the more general mapping (13) from nonzero quaternions of any length (non-unit quaternions) to rotation matrices. Starting from (13) we can derive the body-frame angular velocity $\boldsymbol{\omega}$ of a rotating reference frame $\mathbf{R}$ in terms of the corresponding non-unit quaternion derivatives as follows. From (1) we have

$$[\boldsymbol{\omega}]_\times = \mathbf{R}^T\dot{\mathbf{R}} \quad (16)$$

In terms of any nonunit quaternion which generates $\mathbf{R}$ via (13), this simplifies to

$$\boldsymbol{\omega} = \frac{2}{\|\mathbf{q}\|^2}\mathbf{W}\dot{\mathbf{q}} \quad (17)$$

where $\mathbf{W} \in \mathbb{R}^{3\times 4}$ is

$$\mathbf{W} = \begin{bmatrix} -q_1 & q_0 & q_3 & -q_2 \\ -q_2 & -q_3 & q_0 & q_1 \\ -q_3 & q_2 & -q_1 & q_0 \end{bmatrix} \quad (18)$$

Since we seek to determine the elements of $\dot{\mathbf{q}}$, this is an underdetermined set of three linear equations in four unknowns. Given $\mathbf{q}$, there are infinitely many solutions for $\dot{\mathbf{q}}$ that correspond to the angular velocity $\boldsymbol{\omega}$. The particular solution which minimizes $\dot{\mathbf{q}}^T\dot{\mathbf{q}}$ is given by

$$\dot{\mathbf{q}} = \frac{||\mathbf{q}||^2}{2}\mathbf{W}^\dagger\boldsymbol{\omega} \quad (19)$$

where $\mathbf{W}^\dagger = \mathbf{W}^T(\mathbf{W}^T\mathbf{W})^{-1}$ is the Moore-Penrose pseudo-inverse of $\mathbf{W}$. This further simplifies to

$$\dot{\mathbf{q}} = \frac{1}{2}\mathbf{W}^T\boldsymbol{\omega} = \frac{1}{2}\boldsymbol{\Omega}\mathbf{q} \quad (20)$$

where

$$\boldsymbol{\Omega} = \begin{bmatrix} 0 & -\omega_1 & -\omega_2 & -\omega_3 \\ \omega_1 & 0 & \omega_3 & -\omega_2 \\ \omega_2 & -\omega_3 & 0 & \omega_1 \\ \omega_3 & \omega_2 & -\omega_1 & 0 \end{bmatrix} \quad (21)$$

Remarkably, this is exactly the same formula as the derivative of a unit quaternion. We can also observe that if $\boldsymbol{\omega} \in \mathbb{R}^3$ is considered as a pure quaternion, then $\boldsymbol{\Omega} = \mathbf{Q}(\boldsymbol{\omega})$. The full solution set of (17) can be characterized as this minimum norm solution plus any linear combination of vectors in the kernel (nullspace) of $\mathbf{W}$. The kernel of $\mathbf{W}$ consists of a single vector which turns out to be $\mathbf{q}$ itself. Thus, the solution set of (17) is

$$\dot{\mathbf{q}} = \frac{1}{2}\boldsymbol{\Omega}\mathbf{q} + c\mathbf{q} \qquad \forall c \in \mathbb{R} \quad (22)$$

The quaternion derivative can be similarly derived in terms of spatial angular velocity.

III. Numerical Integration

Since we have characterized the derivative of a non-unit quaternion (22) consistently with the redundant mapping from non-unit quaternions to rotation matrices in (13), we can now exploit this result to numerically integrate rotations using standard, explicit, vector-space methods without regard for staying on any constraint manifold. While quaternion length is still an invariant of (22) when $c = 0$, we no longer require a numerical method to preserve the length if we use the non-unit rotation operator (10) or matrix (13) since the differential differential equation (22) is consistent with the general formulas (10) and (13) regardless of quaternion length.

While we have eliminated the requirement of maintaining unit length, we should note that (10) and (13) were only defined for nonzero quaternions. For quaternion magnitudes near zero, (10) and (13) may be numerically sensitive. Similarly, floating point overflow could also cause inaccuracy for extremely large quaternion magnitudes. We could potentially limit the norm drift by setting the free parameter c as

$$c = k(1 - \mathbf{q}^T\mathbf{q}) \quad (23)$$

where k is some control gain. Since we only want to bound the drift away from extremes, and do not wish to introduce stiff dynamics (by introducing a large effective spring constant), k can be easily selected by starting with a very small value and increasing until an effect is seen over some trial integration length. We use $k = 0.1$ in the benchmark simulation problem below in order to see the effect over a short time span, but much lower values of k would be sufficient. Alternatively, instead of controlling the norm drift we could simply apply a single normalization in the event that the norm drifts too close to zero or the floating point limit. But this is not expected to be necessary very often because the drift is only due to truncation error of the integration method.

A. Relationship to Step-Wise Normalization

In the common simple case where the angular velocity function $\boldsymbol{\omega}(t)$ does not depend on the rotation described by $\mathbf{q}$, we expect the non-unit approach with $c = 0$ and the

step-wise normalization approach to produce identical rotation matrices over time when using any explicit, linear, single-step, multi-stage numerical integration method (i.e. Runge-Kutta methods). We can demonstrate this by induction as follows:

Observe that since $\boldsymbol{\omega}$ is not a function of $\mathbf{q}$, the derivative (22) is linear in $\mathbf{q}$. It follows that $\mathbf{q}_{n+1}$ will be linear in $\mathbf{q}_n$ for any explicit, linear, single-step integration method. Furthermore, if $c = 0$, then $\mathbf{q}_{n+1} = \mathbf{M}\mathbf{q}_n$ for some matrix $\mathbf{M}$ not dependent on $\mathbf{q}_n$. Now suppose two quaternions $\mathbf{q}_n$ and $\mathbf{p}_n$ correspond to the same rotation. Then $\mathbf{p}_n = \alpha\mathbf{q}_n$ for some scalar α. If we compute a single integration step with (22) and $c = 0$ (which is correct for both unit and non-unit quaternions), then $\mathbf{p}_{n+1} = \mathbf{M}\mathbf{p}_n = \alpha\mathbf{M}\mathbf{q}_n = \alpha\mathbf{q}_{n+1}$. Suppose we normalize $\mathbf{p}_{n+1} \leftarrow \mathbf{p}_{n+1}/\|\mathbf{p}_{n+1}\|$ and do nothing to $\mathbf{q}_{n+1}$. Then $\mathbf{p}_{n+1} = \beta\mathbf{q}_{n+1}$ for some scalar β, and $\mathbf{q}_{n+1}$ and $\mathbf{p}_{n+1}$ correspond to the same rotation. Therefore, if $\mathbf{p}_0 = \mathbf{q}_0$, then $\mathbf{p}_N$ and $\mathbf{q}_N$ correspond to the same rotation after N Runge-Kutta steps even if $\mathbf{p}$ is normalized after every step while the length of $\mathbf{q}$ is allowed to drift arbitrarily.

Thus for simple integration of known angular velocity functions, step-wise normalization is not actually needed, but it does not introduce angular error either. However, if the task involves a set of equations where $\boldsymbol{\omega}$ is a function of $\mathbf{q}$, or if we choose $c \neq 0$, then $\mathbf{q}_{n+1} \neq M\mathbf{q}_n$, and the non-unit method solution may differ from the step-wise normalized solution. In these cases, intuition suggests that the long-term accuracy of the non-unit method could be slightly better in some cases because any rotations computed during the intermediate function evaluations within a single Runge-Kutta step will be true rotations, whereas step-wise normalization only occurs at the end of the step [7]. However, the overall accuracy will depend on how the error of intermediate rotation calculations in the step-wise normalization method combines with the truncation error of the integration method. In general, we should expect the two methods to differ by an insignificant amount. Our simulation results below for the Lagrange top confirm this.

In terms of convenience, the non-unit approach enables the use of general purpose, adaptive step-size, ODE integration packages and schemes of any order without modification, by simply changing the way rotations are computed when evaluating derivative function. Step-wise normalization requires an ad hoc overwriting of the state variables after every step, which seems inelegant and requires problem-specific modification to an integration package's source code. In addition, while the non-unit method requires a few more floating point multiplications per integration step (depending on the integration routine), it does not ever require a square root which is a more complex operation despite efficient iterative algorithms [22].

B. Common Exponential Update Methods

Another common practice in robotics is to perform the integration update step by quaternion multiplication with a unit quaternion generated by exponentiating the angular rotation vector (a pure quaternion),

$$\mathbf{q}_{n+1} = e^{\frac{1}{2}\boldsymbol{\omega}\Delta t}\mathbf{q}_n \tag{24}$$

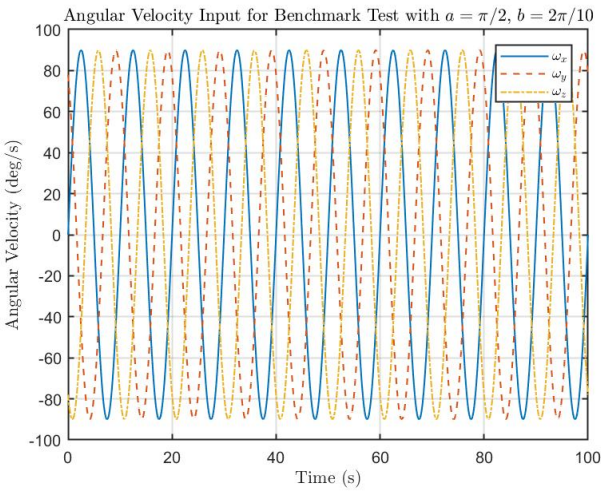

Fig. 1. The body-frame components of the angular velocity vector for the benchmark integration test.

where the exponential of any pure quaternion $\mathbf{v}$ is a unit quaternion given in vector form by

$$e^{\mathbf{v}} = \begin{bmatrix}\cos\|\mathbf{v}\| & \operatorname{sinc}\|\mathbf{v}\|\mathbf{v}^T\end{bmatrix}^T \tag{25}$$

where sinc() is the "sine cardinal" function defined as

$$\operatorname{sinc}(x) \equiv \left\{\begin{array}{ll} 1 & \text{if } x = 0 \\ \frac{\sin x}{x} & \text{otherwise}\end{array}\right\} \tag{26}$$

In rotation matrix notation, the above update is equivalent to

$$\mathbf{R}_{n+1} = \mathbf{R}_n e^{[\boldsymbol{\omega}]_\times \Delta t} \tag{27}$$

where the matrix exponential produces a rotation matrix given by Rodrigues' formula [2], [5]. This approach essentially freezes the angular velocity with a zero-order hold, and then applies the analytical solution of the resulting constant-coefficient linear ODE over the timestep. This strategy is the simplest possible first-order exponential integrator, and it has the clear advantage of preserving unit quaternion length (or rotation matrix orthogonality) at least to roundoff precision. If the angular acceleration $\dot{\boldsymbol{\omega}}$ is available, then a second-order exponential update [6], [9] is given by

$$\mathbf{q}_{n+1} = e^{\frac{1}{2}\boldsymbol{\omega}\Delta t + \frac{1}{2}\dot{\boldsymbol{\omega}}\Delta t^2}\mathbf{q}_n \tag{28}$$

In contrast with these common exponential update methods, a non-unit quaternion integration approach never requires evaluation of trig functions and does not require formula substitutions when the angular velocity is near zero. Furthermore, the non-unit approach enables the use of packaged higher-order integration routines with no special effort or distinction between the way rotations and vector-space variables are treated.

C. Benchmark Comparison of Quaternion-Based Methods

We will now demonstrate the non-unit quaternion method in the numerical integration of a benchmark test problem where

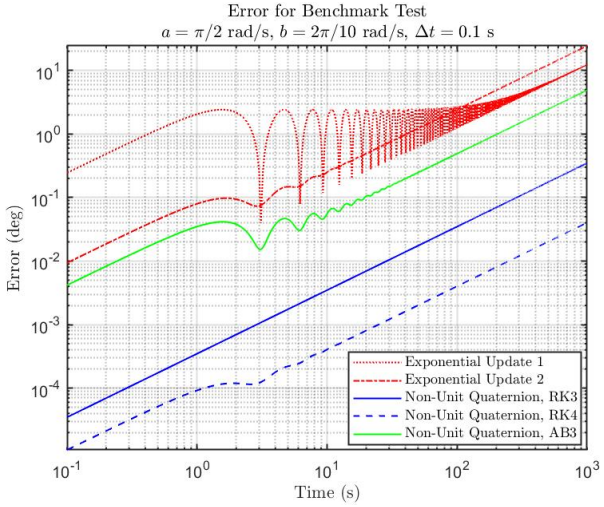

Fig. 2. Log-log plot of the angular error growth over time for five methods. All Non-Unit Quaternion methods used $c = 0$. The first exponential update uses a zero-order hold on angular velocity. The second exponential method involves $\dot{\boldsymbol{\omega}}$.

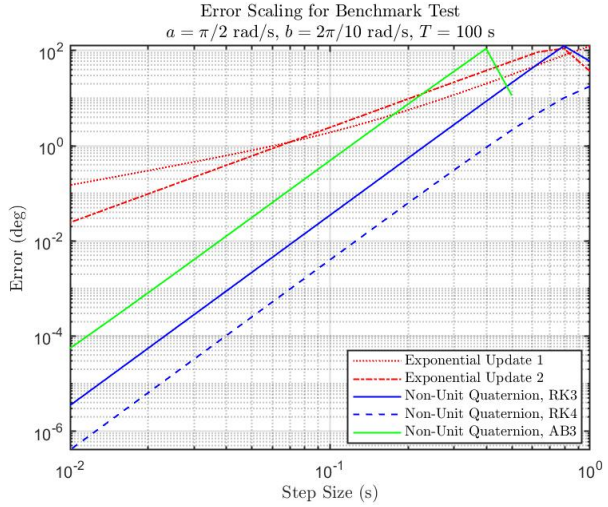

Fig. 3. Log-log plot of angular error at $t = 100$ s versus the integration step size Δt for five methods. The first exponential update (zero-order hold) exhibits first-order error scaling with step size. The second exponential method exhibits second order error behavior. The Runge-Kutta and Adams Bashforth methods all appear to exhibit fourth order behavior for this problem. The kinks in the top right of the plot are due to angular wrap-around, i.e. there is an upper limit of 180 degrees on the angular error by definition.

the body-frame angular velocity vector consists of known sinusoidal functions of time as follows

$$\boldsymbol{\omega}(t) = a \begin{bmatrix} \sin(bt) \\ \sin(bt + 2\pi/3) \\ \sin(bt + 4\pi/3) \end{bmatrix} \tag{29}$$

with $a = \pi/2$ rad/s, $b = 2\pi/10$ rad/s, which is plotted in Figure 1. For validation, an accurate ground-truth solution was first obtained using classic fourth-order Runge-Kutta with step-wise normalization and a time step of $\Delta t = 0.01$ seconds (taking 1000 steps per sinusoidal cycle). For any given method, the rotational error at time t is calculated from the computed rotation matrix $\mathbf{R}(t)$ and the ground-truth rotation matrix $\mathbf{R}_{gt}(t)$ using the matrix logarithm and Frobenius norm as

$$e_R(t) = \left\| \frac{\sqrt{2}}{2} \log(\mathbf{R}^T(t)\mathbf{R}_{gt}(t)) \right\|_F \frac{180^\circ}{\pi} \tag{30}$$

This gives the total angular error between the two rotation matrices in degrees.

First, we can verify our above demonstrated claim that the rotations generated by the non-unit approach agree with the step-wise normalization approach when using a Runge-Kutta method to integrate known angular velocities. Integrating the benchmark problem over 100 seconds at a step size of 0.2 seconds, the maximum difference between the rotation solutions produced by the two methods is on the order of 10^{-13} degrees (which can be attributed to round-off error) despite the significant quaternion norm drift allowed by the non-unit method (33% at $t = 100$ seconds). This same result was also obtained at a much larger time step of 1 seconds despite a norm drift of 96%.

Secondly, we compare the non-unit approach (using various integration schemes with $c = 0$) to the two exponential update methods described above (1 - the zero-order hold on angular velocity (24), and 2 - the method involving $\dot{\boldsymbol{\omega}}$ (28)). The resulting angular errors over time are captured in

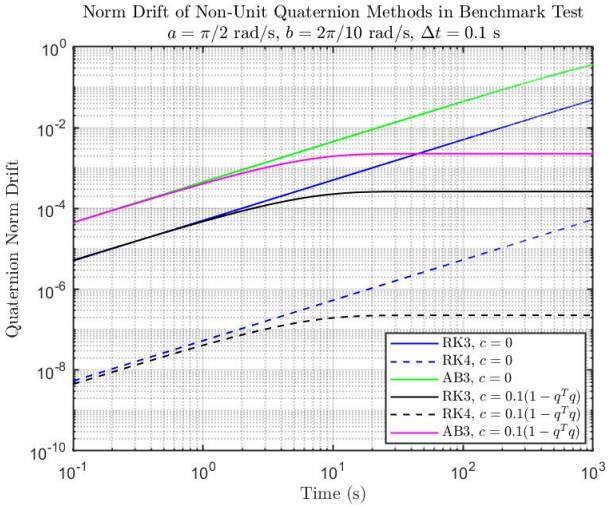

Fig. 4. Log-log plot of quaternion norm drift over the simulated time for non-unit quaternion methods with and without norm control. Since the quaternion does not need to be unit, the goal of norm control is simply to avoid very large or small quaternion magnitudes, and we can use a small gain to avoid stiff dynamics. Alternatively, one can employ a single normalization whenever the quaternion norm becomes unacceptably small.

the log-log plot of Figure 2 for a step-size of 0.1 seconds. Both exponential methods are less accurate than the non-unit implementations using third and fourth order Runge-Kutta (RK3 and RK4), and the third-order Adams-Bashforth multi-step method AB3. We note that tince AB3 is a multi-step method with a single stage, it requires the same number of function evaluations as the exponential methods and is thus amenable to integration of discretely sampled angular velocity data.

Third, we investigate in Figure 3 how the accuracy of all these methods is influenced by the integration step size Δt. The exponential integrators are first- and second-order, so we

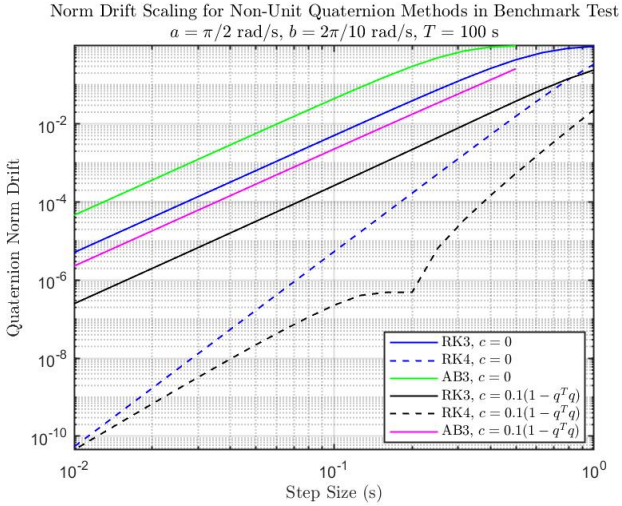

Fig. 5. Log-log plot of norm drift versus step size for the non-unit quaternion methods. The drift exhibits the expected order of the methods, and norm control limits drift regardless of integration step size.

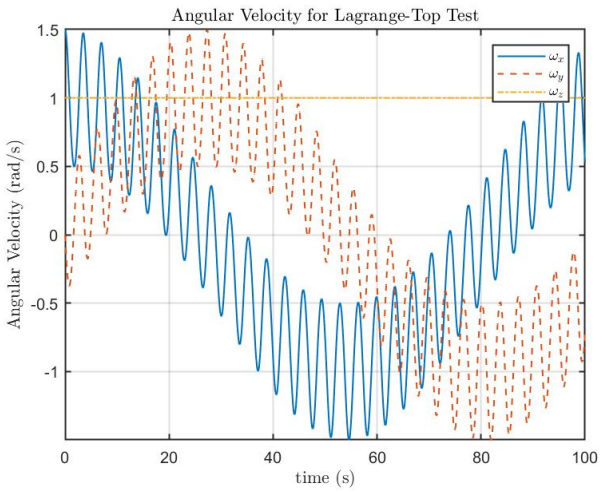

Fig. 6. The body-frame components of the angular velocity vector for the Lagrange-top test.

expect them to eventually perform worse than the third- and fourth-order AB and RK methods as the step size decreases. However, exponential integrators should perform extremely well (even better than the higher-order methods) for problems with relatively constant angular velocities (small $\dot{\boldsymbol{\omega}}$), since they solve the local linearized problem exactly and thus yield the exact solution in the case of constant $\boldsymbol{\omega}$, whereas an RK method would still be approximate for constant $\boldsymbol{\omega}$. In this particular example, the angular accelerations are large enough that the accuracy of the exponential methods is lower than the RK and AB non-unit methods for most time steps. The plot also shows the expected first and second-order convergence behavior of the two exponential methods (evidenced by their log-log slopes of 1 and 2 respectively). The error of the second order exponential method is actually greater than the first order method for timesteps larger than 0.1 seconds (100 steps per cycle). Surprisingly, the third-order methods RK3 and AB3 actually exhibit fourth-order convergence behavior (log-log slope of 4) for this problem. At all time steps, the RK4 error is roughly a factor of ten smaller than the RK3 error, which is itself roughly a factor of ten smaller than the AB3 error.

Finally, we show the effect of quaternion norm control (with $k = 0.1$) over time in Figure 4, and with respect to step size in Figure 5. The norm control successfully limits the norm drift in all cases, while having an insignificant effect on angular accuracy. The maximum norm drift is also affected by step size and behaves according to the order of the methods as shown in Figure 5. Note that the magnitude of the drift is somewhat irrelevant, and k was here chosen so that an effect would be clearly seen within the simulated time scale. Since the quaternion does not need to be unit, the goal of norm control is only to avoid extremely large or small quaternion magnitudes, and a smaller value of k could have been chosen. Note again that in lieu of norm control, one can employ a single normalization whenever the quaternion norm becomes extremely large or small.

IV. Lagrange-Top Test

To further parse the difference between the proposed non-unit approach, step-wise normalization, and other non-quaternion methods for representing orientation, we consider a case where the derivative of the angular velocity itself involves a rotation calculation. A Lagrange top is a rigid body with inertia symmetry about the z axis ($I_x = I_y$). While rotating, a single point on the z axis of the body is fixed in space, and the body's center of mass lies on the z axis a distance of h away from the pivot point. The equations of motion are

$$\begin{aligned} \dot{\mathbf{R}} &= \mathbf{R}[\boldsymbol{\omega}]_\times \\ \dot{\boldsymbol{\omega}} &= \mathbf{I}^{-1}\left(-[\boldsymbol{\omega}]_\times \mathbf{I}\boldsymbol{\omega} + mgh[\mathbf{e}_3]_\times \mathbf{R}^T \mathbf{e}_3\right) \end{aligned} \tag{31}$$

where $\mathbf{e}_3 = [0\ 0\ 1]^T$, m is the mass, g is the acceleration of gravity in the global z direction, and $\mathbf{I}$ is the body-frame moment of inertia matrix about the pivot point. For this simulation, we use $\mathbf{I} = diag(1, 1, 0.5)$ Nm^2, $\omega(0) = [1.5\ \ 0\ \ 1]^T$ rad/s, $m = 1$ kg, $g = 1$ m/s^2, $h = 1$ m, an initial rotation of identity, and an integration length of 100 seconds. As shown in Figure 6 the angular velocity about the body-frame z axis remains constant while the x and y components oscillate out of phase with two distinct modes.

As in the previous section, we computed a ground-truth simulation with step-wise normalized RK4 at a step size of $\Delta t = 0.001$ seconds. We then evaluated the angular error of solutions using non-unit quaternions, step-wise normalized quaternions, XYZ Euler angles, and rotation matrices with step-wise Gram-Schmidt orthogonalization, each integrated with RK4 and $\Delta t = 0.1$ seconds. The resulting semi-log plot of the error over time is shown in Figure 7. The quaternion methods are the most accurate in the long-term, and we see that the Euler angle error undergoes a sharp increase due to proximity to a singularity. As expected from our discussion in Section III-A, the non-unit quaternion formulation exhibits similar accuracy to the step-wise normalization approach. We further examine the error behavior with respect to step size in Figure 8. The quaternion methods are more accurate at

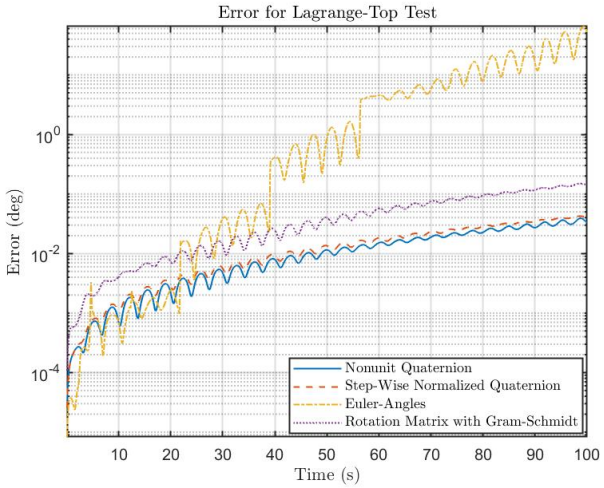

Fig. 7. Angular error over time for the four tested methods.

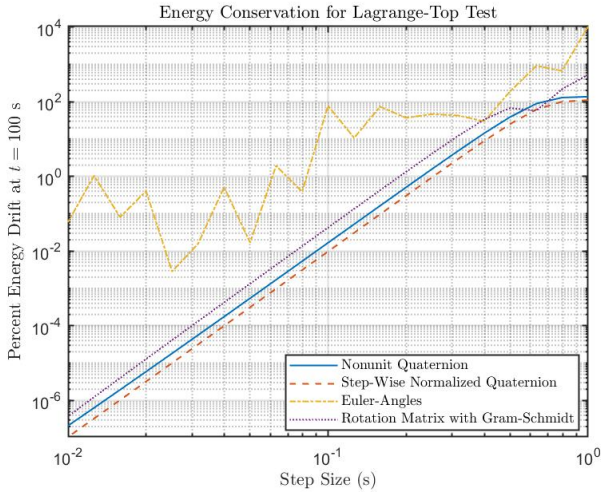

Fig. 9. Energy conservation scaling with respect to step size.

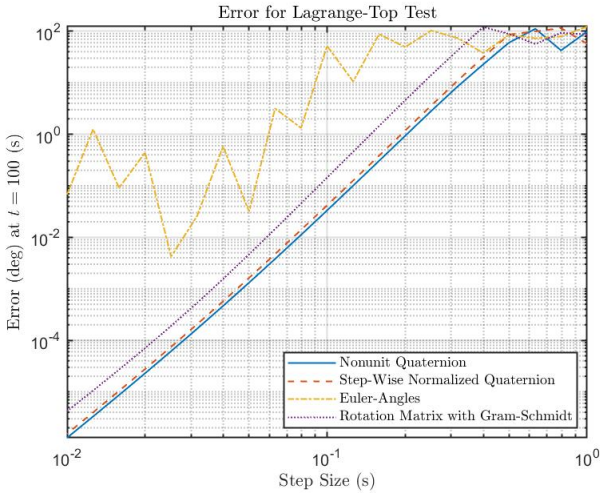

Fig. 8. Angular error scaling with respect to step size.

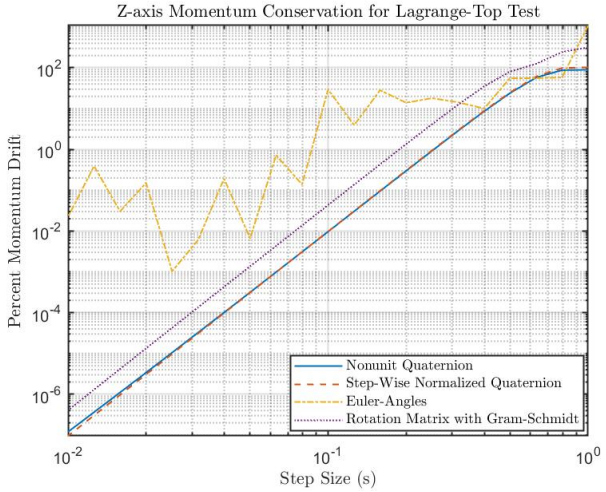

Fig. 10. Momentum conservation scaling with respect to step size.

most step sizes, and the Euler angle approach shows irregular scaling behavior due to the singularity issue.

In addition to the structure of $\mathrm{SO}(3)$, the Lagrange-top equations have two scalar invariants that may be desirable to preserve over long term numerical integration: total energy and angular momentum about the global z axis. Since the non-unit quaternion approach is only designed to preserve the $\mathrm{SO}(3)$ structure, we do not expect it by itself to automatically conserve other invariants when using just any standard numerical integration scheme. Still, we examine in Figures 9 and 10 the performance of the different rotation parameterizations with respect to conservation of invariants using RK4 (which is known to be dissipative). The general trend of the conservation scaling with stepsize parallels the angular error behavior. Energy conservation with step-wise normalization appears just slightly better than with non-unit quaternions, perhaps because some of the energy dissipation inherent to RK4 is being offset by small approximation errors in the function evaluations between normalizations. There is no significant difference in momentum conservation between non-unit and step-wise normalized quaternions. Application of non-unit quaternions to symplectic and energy-conserving integration schemes will be considered in future work.

V. Conclusions

We have derived and demonstrated a formulation that allows consistent integration of rotations via non-unit quaternions without degradation of the $\mathrm{SO}(3)$ structure. This eliminates the need to constrain the quaternion length to unity by the various methods reported in the literature while retaining the advantages of unit quaternions - efficiency and lack of singularities. It turns out that the conventional formula for the derivative of a unit quaternion is the minimum-norm solution for the derivative of a non-unit quaternion. We show that step-wise normalization and the non-unit approach with $c = 0$ will produce equivalent rotations in the case of integrating a known angular velocity function with any Runge-Kutta method. Simulations on a benchmark angular velocity function showed that non-unit quaternions integrated

with standard vector space methods can be superior to two common quaternion integration methods using exponential updates. Simulations of a Lagrange-top showed similar accuracy to step-wise normalization, and increased accuracy over integration of both Euler angles and rotation matrices with Gram-Schmidt orthogonalization. The convenience of the proposed method is expected to be useful in many robotics-related applications highlighted in the introduction, such as simulation, human-body tracking, aerial vehicles, and soft robotics dynamics, as well as in computer graphics, aerospace engineering, physics, and other fields. Future work on non-unit quaternions could include further investigation of numerical convergence properties and computational complexity on other types of problems, algorithms for estimation and control, symplectic integration routines, and further application to the dynamics of soft robot structures and articulated rigid-body trees.

REFERENCES

[1] A. Müller, "Coordinate Mappings for Rigid Body Motions," *Journal of Computational and Nonlinear Dynamics*, vol. 12, no. 2, 2017.

[2] R. M. Murray, Z. Li, and S. S. Sastry, *A Mathematical Introduction to Robotic Manipulation*. Boca Raton, FL: CRC Press, 1994.

[3] E. Hairer, C. Lubich, and G. Wanner, *Geometric Numerical Integration: Structure Preserving Algorithms for Ordinary Differential Equations*. Springer-Verlag, 2006.

[4] J. Park and W.-K. Chung, "Geometric Integration on Euclidean Group With Application to Articulated Multibody Systems," *IEEE Transactions on Robotics*, vol. 21, no. 5, pp. 850–863, 2005.

[5] P. Krysl and L. Endres, "Explicit Newmark/Verlet algorithm for time integration of the rotational dynamics of rigid bodies," *International Journal for Numerical Methods in Engineering*, vol. 62, no. 15, pp. 2154–2177, 2005.

[6] A. Müller, "Approximation of finite rigid body motions from velocity fields," *ZAMM Zeitschrift fur Angewandte Mathematik und Mechanik*, vol. 90, no. 6, pp. 514–521, 2010.

[7] M. S. Andrle and J. L. Crassidis, "Geometric Integration of Quaternions," *Journal of Guidance, Control, and Dynamics*, vol. 36, no. 6, pp. 1762–1767, 2013.

[8] M. Boyle, "The Integration of Angular Velocity," *Advances in Applied Clifford Algebras*, vol. 27, no. 3, pp. 2345–2374, 2017.

[9] S. Farsoni, C. T. Landi, F. Ferraguti, C. Secchi, and M. Bonfe, "Compensation of Load Dynamics for Admittance Controlled Interactive Industrial Robots Using a Quaternion-Based Kalman Filter," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 672–679, 2017.

[10] X. Yun and E. Bachmann, "Design, Implementation, and Experimental Results of a Quaternion-Based Kalman Filter for Human Body Motion Tracking," *Robotics, IEEE Transactions on*, vol. 22, no. 6, pp. 1216–1227, 2006.

[11] L. Jung Keun and E. J. Park, "Minimum-Order Kalman Filter With Vector Selector for Accurate Estimation of Human Body Orientation," *Robotics, IEEE Transactions on*, vol. 25, no. 5, pp. 1196–1201, 2009.

[12] X. Yuan, S. Yu, S. Zhang, G. Wang, and S. Liu, "Quaternion-based unscented Kalman filter for accurate indoor heading estimation using wearable multi-sensor system." *Sensors (Basel, Switzerland)*, vol. 15, no. 5, pp. 10 872–90, may 2015.

[13] K. Feng, J. Li, X. Zhang, C. Shen, Y. Bi, T. Zheng, and J. Liu, "A New Quaternion-Based Kalman Filter for Real-Time Attitude Estimation Using the Two-Step Geometrically-Intuitive Correction Algorithm," *Sensors*, vol. 17, no. 9, p. 2146, sep 2017.

[14] A. Santamaria-Navarro, G. Loianno, J. Solà, V. Kumar, and J. Andrade-Cetto, "Autonomous navigation of micro aerial vehicles using high-rate and low-cost sensors," *Autonomous Robots*, pp. 1–18, dec 2017.

[15] P. Betsch and R. Siebert, "Rigid body dynamics in terms of quaternions: Hamiltonian formulation and conserving numerical integration," *International Journal for Numerical Methods in Engineering*, vol. 79, no. 4, pp. 444–473, jul 2009.

[16] L. J. Seelen, J. T. Padding, and J. A. Kuipers, "Improved quaternion-based integration scheme for rigid body motion," *Acta Mechanica*, vol. 227, no. 12, pp. 3381–3389, 2016.

[17] F. Zhao and B. G. M. Van Wachem, "A novel Quaternion integration approach for describing the behaviour of non-spherical particles," *Acta Mechanica*, vol. 224, no. 12, pp. 3091–3109, 2013.

[18] Z. Terze, A. Müller, and D. Zlatar, "Singularity-free time integration of rotational quaternions using non-redundant ordinary differential equations," *Multibody System Dynamics*, vol. 38, no. 3, pp. 201–225, 2016.

[19] J. Till and D. C. Rucker, "Elastic Stability of Cosserat Rods and Parallel Continuum Robots," *IEEE Transactions on Robotics*, 2017.

[20] I. Tunay, "Spatial Continuum Models of Rods Undergoing Large Deformation and Inflation," *IEEE Transactions on Robotics*, vol. 29, no. 2, pp. 297–307, apr 2013.

[21] J. B. Kuipers and J. B., *Quaternions and rotation sequences : a primer with applications to orbits, aerospace, and virtual reality*. Princeton University Press, 1999.

[22] P. Markstein, "Software division and square root using Goldschmidt's algorithms," *In 6th Conference on Real Numbers and Computers*, pp. 146—-157, 2004.

D. Caleb Rucker (M'13) received B.S. degrees in engineering mechanics and mathematics from Lipscomb University, Nashville, TN, in 2006, and the Ph.D. degree in mechanical engineering from Vanderbilt University in 2010, Nashville TN.

From 2011 to 2013, he was a postdoctoral fellow in Biomedical Engineering at Vanderbilt University before joining The University of Tennessee, Knoxville, TN, USA, in 2013 as an Assistant professor of mechanical engineering where he currently directs the Robotics, Engineering, and Continuum Mechanics in Healthcare Laboratory (REACH Lab).

Dr. Rucker received the National Science Foundation CAREER Award in 2017. His current research interests include medical robotics, soft and continuum robots, and dynamic systems.